Robust Image Corner Detection Based on the Chord-to-Point Distance Accumulation Technique

Mohammad Awrangjeb, *Member, IEEE*, and Guojun Lu, *Senior Member, IEEE*

***Abstract*—Many contour-based image corner detectors are based on the *curvature scale-space* (CSS). We identify the weaknesses of the CSS-based detectors. First, the "curvature" itself by its "definition" is very much sensitive to the local variation and noise on the curve, unless an appropriate smoothing is carried out beforehand. In addition, the calculation of curvature involves derivatives of up to second order, which may cause instability and errors in the result. Second, the Gaussian smoothing causes changes to the curve and it is difficult to select an appropriate smoothing-scale, resulting in poor performance of the CSS corner detection technique. We propose a complete corner detection technique based on the *chord-to-point distance accumulation* (CPDA) for the discrete curvature estimation. The CPDA discrete curvature estimation technique is less sensitive to the local variation and noise on the curve. Moreover, it does not have the undesirable effect of the Gaussian smoothing. We provide a comprehensive performance study. Our experiments showed that the proposed technique performs better than the existing CSS-based and other related methods in terms of both *average repeatability* and *localization error*.**

***Index Terms*—Chord-to-point distance accumulation, corner detection, curvature scale-space.**

I. Introduction

IMAGE feature detection and matching are two fundamental problems of computer vision and image processing research. For example, in image copyright protection, some pixels in the original image are used either to calculate the signature [1] or to embed the watermark [2]. However, geometric transformations change the location of those pixels. Consequently, the copyright verifier fails to track the copyright information in the transformed image. In such cases, if the locations of those pixels are defined with respect to the salient features of the image, e.g., corners, the verifier will be able to locate those pixels easily.

In feature detection problem, a set of representative features, most often corners, are detected for all images [3]. In feature matching problem, the representative features of the test image and the stored images are compared to identify the transformed images of the test image [4]. We will focus on the corner detection problem in this paper.

The contour-based corner detectors [3], [5]–[11] are mainly based on the *curvature scale-space* (CSS) (contours can be open or close). They smooth the planar-curves with the Gaussian function at different smoothing-scales. Then they estimate the curvature on each point of the smoothed curves. The absolute curvature maxima points are gathered in the candidate corner set, from which the weak (also called "round" corners in the literature [8]) and false corners (see Section III-D for their characteristics) are eliminated using thresholds. Some CSS corner detectors [3], [8], [9], which detect corners using one or more high smoothing-scales, also follow a corner tracking step in order to improve localization.

The existing CSS corner detectors suffer from two main problems. First, the CSS curvature estimation technique adopted by the existing detectors is highly sensitive to the local variation and noise on the curve. In addition, the curvature estimation involves higher order derivatives of curve point-locations up to second order which cause errors and instability in results. Second, the CSS corner detection technique requires appropriate Gaussian smoothing-scale selection which is a difficult task. Consequently, smoothing with inappropriate Gaussian scales by the existing CSS detectors results in poor corner detection performance.

The purpose of this paper is to propose a new corner detection technique which overcomes the aforementioned problems associated with the existing CSS corner detectors. We present a complete corner detection technique based on the *chord-to-point distance accumulation* (CPDA) for the discrete curvature estimation [12]. The CPDA discrete curvature estimation technique is less sensitive to the local variation and noise on the curve. It does not use any derivative of the curve-point locations at all. Moreover, it does not have the undesirable effect of the Gaussian smoothing. As a result, the proposed CPDA corner detector greatly overcomes the problems associated with the existing CSS corner detectors and offers better performance.

The contribution and organization of this paper are summarized as follows.

- First, we identify and analyze the problems with the existing CSS corner detetcors (Section II-B).
- Second, we propose a complete corner detector based on the CPDA discrete curvature estimation [12]. The performance of the proposed detector is improved not only by the use of CPDA discrete curvature estimation, but also by introducing some techniques to make the detector more robust (Section III).
- Third, we carried out a comprehensive performance study. The proposed CPDA corner detector outperformed the existing most promising detectors [3], [8], [10], [11] in terms of both *average repeatability* and *localization error* (Section IV-C).

Manuscript received June 01, 2007; revised May 12, 2008. Current version published October 24, 2008. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Daniel Gatica-Perez.

The authors are with the Gippsland School of Information Technology, Monash University, Churchill, Vic 3842, Australia (e-mail: Mohammad.Awrangjeb@infotech.monash.edu.au; Guojun.Lu@infotech.monash.edu.au).

Digital Object Identifier 10.1109/TMM.2008.2001384

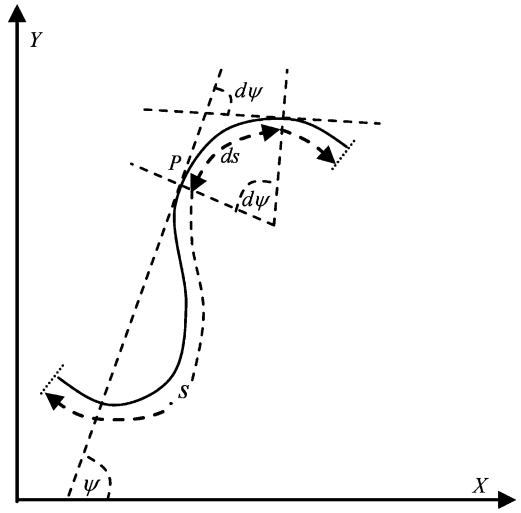

Fig. 1. Intrinsic definition of curvature.

- Finally, we discuss and analyze why the proposed CPDA detector performs better than the existing CSS-based detectors (Section IV-D).

II. Related Work

In this section, we first briefly present the CSS curvature estimation technique and then identify and analyze the problems associated with the existing CSS corner detectors that adopted the CSS curvature estimation technique. Finally, we present the CPDA discrete curvature estimation technique that will be used by the proposed CPDA corner detector to overcome the problems with the existing CSS-based detectors.

A. CSS Curvature Estimation

The curvature κ at a point P of a curve is defined as the instantaneous rate of change of ψ, which is the angle subtended by the tangent at P with the x-axis, with respect to the arc-length s [5] (see Fig. 1),

$$\kappa = \frac{d\psi}{ds}. \tag{1}$$

Let $\Gamma(t) = (x(t), y(t))$ be a curve of length n pixels, where $x(t)$ and $y(t)$ denote the x and y positions of each point on the curve with respect to the arbitrary parameter $t, 1 \leq t \leq n$. The curvature defined in (1) becomes [3], [5]–[11]

$$\kappa(t) = \frac{\dot{x}(t)\ddot{y}(t) - \ddot{x}(t)\dot{y}(t)}{(\dot{x}^2(t) + \dot{y}^2(t))^{3/2}} \tag{2}$$

where $\dot{x}(t)$ and $\dot{y}(t)$ are first and $\ddot{x}(t)$ and $\ddot{y}(t)$ are second order derivatives with respect to t. The first order and second order derivatives at a point P_i on a curve are defined as

$$\dot{P}_i = \frac{P_{i+1} - P_{i-1}}{2} \text{ and } \ddot{P}_i = \frac{\dot{P}_{i+1} - \dot{P}_{i-1}}{2}. \tag{3}$$

We see that only one neighbor point on each side of point P_i is considered to estimate the first order derivative on P_i and only two neighbor points on each side of that point are considered to estimate the second order derivatives. This means that the CSS curvature estimation technique considers a very small neighborhood (2×2) on both sides of each point to evaluate the curvature at that point. This makes the technique very much susceptible to the local variation of the curve and to the noise, as we discuss shortly in the next section.

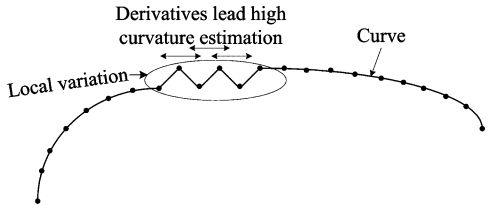

Fig. 2. Local variation of a curve.

Curve smoothing prior to curvature measurement reduces the effect of the local variation and noise. To do that each coordinate functions $x(t)$ and $y(t)$ of the curve are convolved with the Gaussian function $g(t, \sigma)$, where σ denotes the smoothing-scale. The curvature on the smoothed curve $\Gamma_s(t, \sigma) = (x_s(t, \sigma), y_s(t, \sigma))$, becomes [3], [5]–[11]

$$\kappa(\tau, \sigma) = \frac{\dot{x}_s(t, \sigma)\ddot{y}_s(t, \sigma) - \ddot{x}_s(t, \sigma)\dot{y}_s(t, \sigma)}{(\dot{x}_s^2(t, \sigma) + \dot{y}_s^2(t, \sigma))^{3/2}} \tag{4}$$

where

$$\left.\begin{array}{c} \dot{x}_s(t, \sigma) = x(t) * \dot{g}(t, \sigma), \ddot{x}_s(t, \sigma) = x(t) * \ddot{g}(t, \sigma), \\ \dot{y}_s(t, \sigma) = y(t) * \dot{g}(t, \sigma), \text{ and } \ddot{y}_s(t, \sigma) = y(t) * \ddot{g}(t, \sigma), \end{array}\right\} \tag{5}$$

where $*$ denotes the convolution operation. As the convolution operation is associative under differentiation, derivatives of the convolved curve can be calculated using (5) [8].

B. Problems With the Existing CSS Corner Detectors

In this section, we will first analyze the problems with the CSS corner detection technique and then discuss how the existing CSS-based detectors suffer from these problems.

1) Problems: There are two key problems associated with the CSS corner detection technique. The first problem is directly related to the curvature itself defined in (1) and can be understood using (2) and (3). Since the curvature, by definition, means the instantaneous rate of change of angle ψ [see (1)], it is very much sensitive to the local variation and noise on the curve. In a high local variation region, ψ changes significantly from point to point within a short curve segment. As depicted in Fig. 2, in such a small but highly variable curve-region the derivatives of curve point-locations may lead to the high curvature estimation. As a result, the existing CSS-based detectors may detect many weak and false corners, if such local variation or noise is not smoothed away using a high smoothing-scale beforehand. However, smoothing has its own problem as discussed shortly (the second problem).

Though the Gaussian convolution is used to calculate the derivatives of curve point-locations as shown in (5), the precise approximation of the higher order derivatives is still a difficult task and often causes instability and introduces errors [13]. The problem becomes more severe under geometric transformations

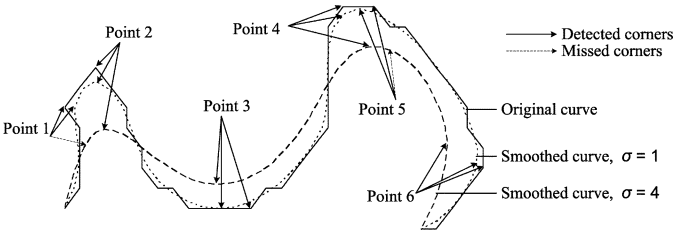

Fig. 3. Effect of curve smoothing using Gaussian function with different smoothing-scale σ. The original curve as shown above is a high resolution version of "curve 7" in Fig. 5(b).

when the curve point-locations are always approximated and, therefore, may not be stable. Consequently, the derivative-based curvature value defined in (4) may change considerably between original and transformed curves. Such unstable and erroneous estimated curvature not only affects the corner detection performance under geometric transformations, but also may result in poor corner matching performance in any of their later applications [4].

The second problem is related to the curve smoothing using the Gaussian function. The aim of the smoothing is to reduce the effect of local variation and noise so as to remove weak and false corners. However, two direct effects of the Gaussian smoothing are: first, it shrinks the curve and second, it smoothes out the details if the smoothing-scale is high. On a smoothed curve, corners are usually detected at those points where the absolute curvature maxima values become higher than the threshold. However, the estimated curvature value at a corner point may decrease below the threshold with smoothing using high σ. The sharp (strong) corners are easily identifiable with smoothing using large σ, their estimated curvature is less accurate and localization is worse than with smoothing using small σ. However, using small σ may detect many weak and false corners. Moreover, one smoothing-scale may not be suitable for all curves. Since the local variation and noise on the curve are unknown, curves of the same length may require different smoothing-scales, even different segments of the same curve may require different smoothing-scales. Therefore, choosing an appropriate Gaussian smoothing-scale for a given curve is a very difficult task.

This above problem is depicted in Fig. 3. The original curve in Fig. 3 is a high resolution version of "curve 7" in Fig. 5(b), where we see three strong corners (points 2, 3, and 4 in Fig. 3). In the high resolution version of the curve (original curve in Fig. 3), we see many weak corners and we choose points 1, 5, and 6 for illustration. Smoothing with both $\sigma = 1$ and 4 can detect all three strong corners; but while smoothing with $\sigma = 1$ detects all the weak corners, smoothing with $\sigma = 4$ misses two weak corners (points 1 and 5). However, corner localization is better with lower smoothing-scale (see locations of point 3 in original and smoothed curves in Fig. 3). Note that the detected corners are shown with solid arrows and the missed corners are shown with dotted arrows.

2) Existing CSS-Based Detectors: The first problem discussed above was an inherent problem with all the existing CSS corner detectors [3], [5]–[11], since they used the same CSS curvature estimation technique discussed in Section II-A. In the rest of this section, we will discuss how the existing CSS-based detectors suffer from the second problem.

The earliest CSS corner detector by Rattarangsi and Chin [5] and its improved version [6] detected corners in the complete scale-space by considering all possible smoothing-scales so as to overcome the second problem. However, the tree construction and parsing using the complete scale-space resulted in high computational complexity. The adaptive smoothing technique in [7] reduced the computational complexity by eliminating the construction of the complete scale-space map and by avoiding the tree representation of the scale-space. However, the adaptive smoothing technique itself incurred additional computational cost which was supposed to be compensated by the slightly improved robustness.

Mokhtarian and Suomela [8] detected corners at a high scale and tracked them through multiple lower scales to the lowest scale in order to improve localization. On the one hand, when corners were detected in a very high scale, this detector showed high robustness but lost many strong corners which might require lower smoothing-scales to be detected. On the other hand, if corners were detected in a low scale it introduced many weak corners which might require higher smoothing-scales to be removed. As a result, this detector also suffered from the second problem. Many of its improved versions (e.g., [3], [9]), which selected the smoothing-scales based on the curve-length, also failed to overcome this problem. Because even curves of the same length may require different smoothing-scales depending on the level of local variation and noise as discussed above.

The *multi-scale curvature product* (MSCP) detector [10] and the single-scale detector [11] tried to overcome the second problem to a great extent. They compensated the risk associated with possible inappropriate smoothing-scale selection by adopting different strategies. The MSCP detector used the curvature product of three estimated curvature values at each point using three smoothing-scales. As a result, in terms of curvature product, the strong corners became more distinguishable from the weak corners. He and Yung [11] used the adaptive curvature-threshold and the dynamic region-of-support on both sides of each curvature extremum point. Consequently, both of them offered better corner detection performance than many of the aforementioned detectors. In this paper, we will compare most promising detectors [3], [8], [10], [11] with our proposed corner detector (see Section IV).

C. CPDA Discrete Curvature Estimation

The previously proposed single chord-to-point distance measurement technique [14] was not reliable, since such distance depends upon the location of the chord [12]. In order to increase the reliability, Han and Poston [12] proposed the *chord-to-point distance accumulation* (CPDA) technique for measuring the discrete curvature. A chord is moved along a curve and the perpendicular distances from each point on the curve to the chord are summed to represent the curvature. In contrast to the conventional CSS curvature estimation technique adopted by the existing CSS corner detectors, the CPDA technique is completely based on the Euclidean distance and does not involve any derivative of the curve-point locations at all.

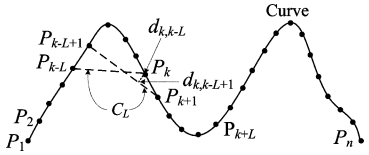

Fig. 4. Chord-to-point distance accumulation technique for a chord C_L of length L. In the literature [12], here chord-length L denotes the arc-length of the interior curve-segment. For example, in this Fig. $L = 5$.

The formal definition of the CPDA discrete curvature estimation technique is as follows. Let $P_1, P_2, P_3, \ldots P_n$ be the n points of the parameterized curve $\Gamma(t) = (x(t), y(t)), 1 \leq t \leq n$, as shown in Fig. 4. To measure the curvature $h_L(k)$ at a point P_k using a chord C_L of length L, we move the chord on each side of P_k at most L points while keeping P_k as an interior point. Note that according to [12], the chord-length L denotes the arc-length of the interior curve-segment as shown in Fig. 4. Starting the movement from the point P_{k-L} when the two ends of the chord are at P_{k-L} and P_k respectively, we measure the perpendicular distance $d_{k,k-L}$ from P_k to the chord. Then we move the chord one point ahead when its two ends are at P_{k-L+1} and P_{k+1} respectively and measure the perpendicular distance $d_{k,k-L+1}$. The procedure continues by moving the chord one point at a time and stops when the both ends of the chord move to the points P_k and P_{k+L} respectively. Then we accumulate all distances to calculate the CPDA discrete curvature at the point P_k as

$$h_L(k) = \sum_{j=k-L}^{k} d_{k,j}. \tag{6}$$

Since $d_{k,k-L} = d_{k,k} = 0$, the above formula is simplified to

$$h_L(k) = \sum_{j=k-L+1}^{k-1} d_{k,j}. \tag{7}$$

In general, the above CPDA function has the following advantages over the CSS [12]. First, it does not shrink the curve and smooth out the details. Second, the estimated CPDA curvature increases with the increase of L and the detected features (curvature zeros and extrema) are quite stable for a wide range of L. Since the chord smoothing does not change the curve-point locations, the details of the curve are not physically smoothed out when L increases. Though small and large L values are not good, still a selection of medium L values helps detecting the strong corners.

The proposed CPDA corner detector based on the above discrete curvature estimation technique does not suffer from the key problems associated with the existing CSS corner detectors, because the CPDA discrete curvature estimation does not directly implement the "curvature definition" and it does not require the appropriate σ value selection. We will detail this effect later in Section IV-D.

III. Proposed CPDA Corner Detector

The proposed corner detector first extracts planar curves from the edge image detected by the Canny edge detector [15]. Each curve is then smoothed with a small width Gaussian kernel in order to remove quantization noise and trivial details. Please note that this smoothing is done only once as a preprocessing. We use the CPDA technique [12] to estimate the curvature on the smoothed curves. In order to make strong and weak corners more distinguishable, we first use three chords of different lengths to estimate three normalized discrete curvature values on each point of the smoothed curve. Then we multiply the normalized curvatures to obtain the curvature product (a single estimated curvature) at each point. The maxima of the absolute curvature products along the smoothed curve are then obtained as candidate corners. Finally, it follows a two-step refinement process that uses a curvature-threshold and an angle-threshold to remove weak and false corners, respectively.

The outline of the proposed corner detector is as follows.

- Find the edge image using the Canny edge detector.
- Extract edges (curves) from the edge image:
  — fill gaps if they are within a range and select long edges,
  — find T-junctions and mark them as T-corners.
  — obtain the "status" of each selected edge Γ as either "loop" or "line."
- Smooth Γ using a small width Gaussian kernel in order to remove quantization noises and trivial details. This small scale Gaussian smoothing also offers good localization of corners.
- At each point of the smoothed curve Γ_s, compute three discrete curvatures following the CPDA technique using three chords of different lengths.
- Find three normalized curvatures at each point of Γ_s and then multiply them to obtain the curvature product.
- Find the local maxima of the absolute curvature products as candidate corners and remove weak corners by comparing with the curvature-threshold T_h.
- Calculate angles at each candidate corners obtained from the previous step and compare with the angle-threshold δ to remove false corners.
- Find corners, if any, between the ends of smoothed "loop" curves and add those corners which are far away from the detected corners.
- Compare T-corners with the detected corners and add those T-corners which are far away from the detected corners.

We will detail the proposed CPDA corner detection technique in the following subsections. All the chosen parameter values that we will present below were decided either from the existing work or based on our empirical study (see Section IV-C1 for detail).

A. Edge Extraction and Selection

In the Canny edge detector [15], too many weak and noisy edges will be detected when the edge strength threshold is set too low. Conversely, many legitimate edges will be missed when the threshold is set too high. In addition, the edge strength may be changed due to geometric transformations. Therefore, in our implementation, we choose the Canny edge detection thresholds as $low = 0.2$ and $high = 0.7$, after experimentation.

Some of the edges detected as above may be quite short and may result from strong noises. It is difficult to smooth short edges due to the restriction of the smoothing window size

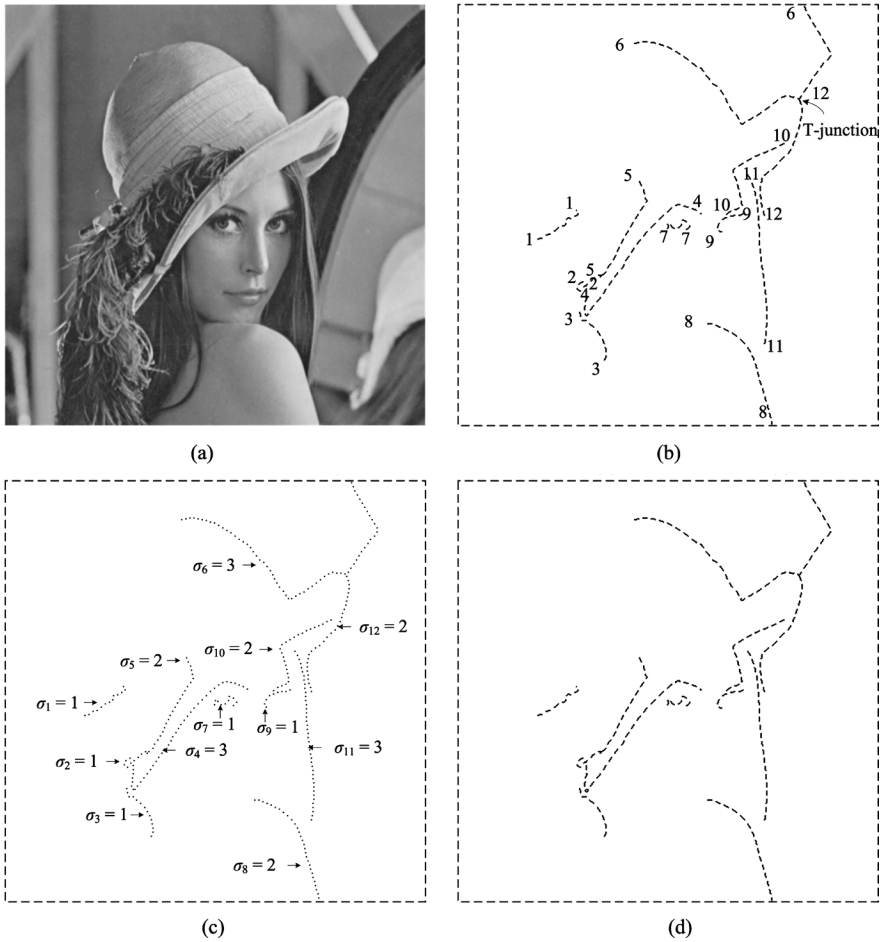

Fig. 5. Edge detection and smoothing: (a) original "Lena" image (512 × 512); (b) edge image from (a) using Canny edge detector with thresholds $low = 0.2$ and $high = 0.7$, total 12 edges were detected and are shown here where both ends of each edge is numbered for easy identification; (c) Gaussian smoothed curves with small σ's to reduce the effect of noise; and (d) superposition of (b) and (c) to show small scale Gaussian smoothing does not affect localization much.

(window size should be smaller than the edge length). Thus in our implementation, edges are selected only when their lengths n meet the following condition:

$$n > \frac{width + height}{\alpha} \tag{8}$$

where *width* and *height* are width and height of the image and α is a length control parameter. If α is small, only long edges are selected; if α is large, short edges are also selected. In our experiments, we set $\alpha = 25$ [11]. Therefore, for an image of size 512×512, the minimum length of the selected curves is $n_{\min} = 42$. The above edge extraction setup avoids selecting large number of weak and very short edges which may not contain strong corners, thereby helps speeding up the later steps.

If an edge runs through any point which is within 2 pixels away from an end of another edge [8], we select that end as a T-junction and add to the set T-corners. See Fig. 5(b) for the extracted edges from "Lena" image using the above setup, where a T-corner is detected in the intersection of curves 6 and 12.

We also obtain the "status" of each selected edge Γ as either "loop" or "line." If both ends of a curve are within 5 pixels away [11], the curve is a "loop" curve, otherwise it is a "line" curve. This curve status definition will help determining the corner, if any, between the ends of a loop curve (see Section III-E).

B. Curve Smoothing

The purpose of curve smoothing before curvature estimation is to reduce the effect of noise which may affect the curvature estimation severely. Noise may also be introduced by the edge detector. In general, short edges should be smoothed with small σ and long edges should be smoothed with large σ [9], because smoothing a short edge with a large σ may smooth out important features and smoothing a long edge with a small σ may leave a lot of noises on it. However, σ value depends on the amount of noise which is unknown. Therefore, determining appropriate σ for a given curve of any length is a difficult task.

Since we do not follow any corner tracking step after their detection and large scale smoothing may smooth out many important features, we use small scale Gaussian smoothing to keep the effect of localization problem minimum. For curves of length $n \leq 100$ we select scale $\sigma = 1$; for $100 < n \leq 200$ we select $\sigma = 2$; and for $n > 200$ we select $\sigma = 3$. Fig. 5(c)–(d) show that using such small scale smoothing functions does not change the corner location much. However, this small scale smoothing may not remove all weak and false corners which we will remove by following a two-step refinement process later in Section III-D.

After smoothing, we extend the both ends of the smoothed curve. In the case of a "loop curve," we extend each end by

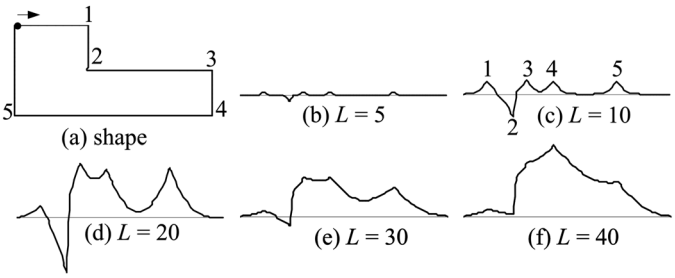

Fig. 6. Distance accumulation technique: (a) original shape (a dot represents start point and arrow represents the direction); (b)–(f) estimated curvature functions using different chord-lengths L.

taking into account of the points of the other end. And in the case of a "line curve," we extend each end by taking into account of the points of the same end. This curve extension prevents missing any prominent corners near to the ends of the curve. The reason is, on points near to the ends, the number of chord movements becomes very low (zero at the ends) during the CPDA discrete curvature calculation using (7). This phenomena results in low curvature estimation for a prominent corner near to any end of the curve. The above curve extension reduces this effect. Note that in the following section we do not calculate curvatures on the points of the extended parts, but we use the points on the extended parts to calculate the discrete curvature values on points near to the ends of the original smoothed curve.

C. Curvature Estimation

We use (7) in Section II-C to calculate the CPDA discrete curvature value on each point of the smoothed curve. One important property of the CPDA discrete curvature estimation is that the curvature value on a point increases with the increase of chord-length L. However, it may be difficult to distinguish different important features with a large L. For example, Fig. 6 shows the curvature estimation using (7). When L is medium say 10 [see Fig. 6(c)], the corners (curvature extrema points) are easily identifiable. But when L increases, say 30 or 40 [see Fig. 6(e)–(f)], many corners become flatter and more indistinguishable.

In general, we need to use large L's for long curves and small L's for short curves. However, choosing a single chord-length for a given curve is difficult, because curves of the same length may contain different types of corners. For instance, when a single chord is used, the corner detector will be sensitive to noise if the chord-length is set too small, but it will smooth out the details of the curve if the chord-length is set too high. Therefore, we calculate three discrete curvature values at each point using three chords of different lengths for the following reasons. First, it is difficult to decide one single chord-length for different curves. Second, we use the product of three estimated curvatures to make the strong corners more distinguishable than the weak and false corners.

We choose three chords of medium lengths $L_1 = 10, L_2 = 20$, and $L_3 = 30$ irrespective of the curve-length to measure the CPDA discrete curvature using (7). Note that the above setting of chord-lengths is suitable with the minimum curve-length $n_{\min} = 42$ discussed in Section III-A.

Let $h_1(k), h_2(k)$, and $h_3(k)$, where $1 \le k \le n$, be three curvature function using three chords of lengths L_1, L_2, and L_3 respectively. The absolute values of $h_j(k)$, where $1 \le j \le 3$, may range from zero to a long integer number depending on the types of corners on the curve. Moreover, the estimated curvature value of the same type of corner (for example, a corner with 60° angle) using any of the three chords may be different on different curves. Consequently, a single curvature threshold setting to remove the weak corners from all types of curves becomes problematic (see first refinement step in Section III-D1). To overcome this problem, we normalize the function $h_j(k)$ using (9), so that the discrete curvature values are in the range [0,1].

$$h'_j(k) = \frac{h_j(k)}{\max(h_j)}, \text{ for } 1 \le k \le n \text{ and } 1 \le j \le 3. \quad (9)$$

However, this normalization has an undesirable effect. On straight-line like curve, where there may be no prominent corner, some of the absolute curvature maxima become larger than the curvature-threshold. These maxima points are detected as false corners and may not be removed by the first refinement step. We will use the second refinement step (see Section III-D2) to remove such false corners.

As three normalized curvature values are computed at each point of a curve, we need to find a single feature value to detect corner by comparing with a single curvature threshold. To find it we simply use the following curvature product as the single feature value:

$$H(k) = h'_1(k) \cdot h'_2(k) \cdot h'_3(k), \text{ for } 1 \le k \le n. \quad (10)$$

The use of above curvature product has an additional advantage. As the normalized curvature value of a strong corner should be higher than that of a weak corner on the same curve, by multiplying three estimated curvatures at each point, the strong corners become more distinguishable than the weak corners.

For example, let three normalized curvature values on a strong corner be 0.6, 0.75, and 0.8 respectively and those on a weak corner be 0.1, 0.15, and 0.2 respectively. So the ratios of curvature values of strong and weak corners are 6, 5, and 4 respectively. However, the curvature products of strong and weak corners are 0.036 and 0.003 whose ratio is 120. Fig. 7 shows the curvature functions for "curve 4" of "Lena" (see Fig. 5) for different chord-lengths. It is evident from Fig. 7 that the strong corner-points are more distinguishable in the curvature product function than in the individual curvature functions with different chord-lengths.

A direct impact of curve smoothing before the curvature estimation using (10) is also shown in Fig. 7(d). We see that though the curvature product function may contain a lot of local small peaks without smoothing [see the thin curve in Fig. 7(d)], the effect of noise is reduced if the curve is smoothed prior to the curvature estimation [see thick curve in Fig. 7(d)]. This will help distinguishing the curvature extrema points corresponding to the strong corners unambiguously.

D. Candidate Corner Set Refinement

We gather the local maxima points on the absolute function of $(H(k))$ from all curves in the candidate corner set. A local

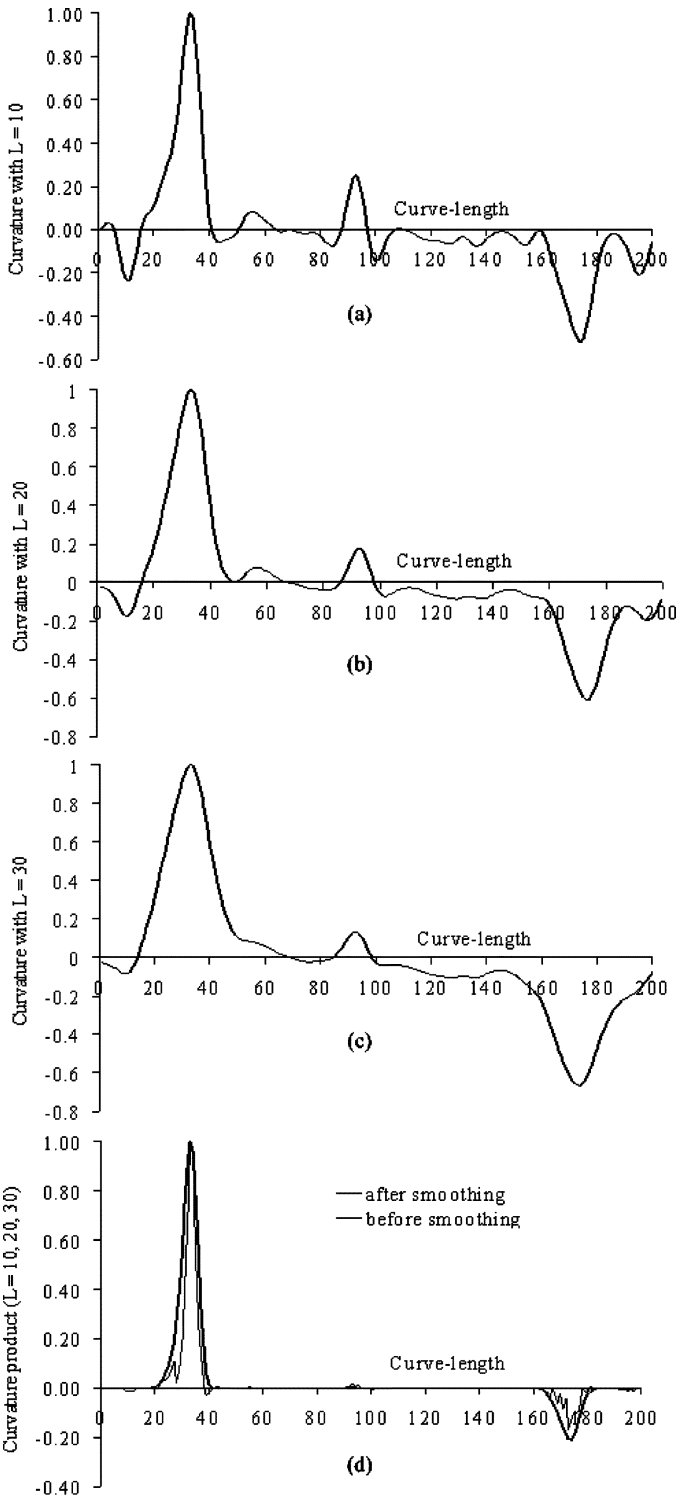

Fig. 7. Normalized CPDA discrete curvature functions for "curve 4" of 'Lena' (see Fig. 5) with different chord-lengths: (a) $L = 10$; (b) $L = 20$; (c) $L = 30$; and (d) the curvature product function with $L = 10, 20$, and 30, where "after smoothing" and "before smoothing" depict $H(k)$ with and without Gaussian smoothing ($\sigma = 3$), respectively, to show the importance of Gaussian smoothing prior to curvature estimation.

maximum is either a strong corner or a weak corner (also called "round" corners in the literature [8]) or a false corner. The later two should not be detected as corners. The strong corners are very sharp in nature and visually prominent features on curves.

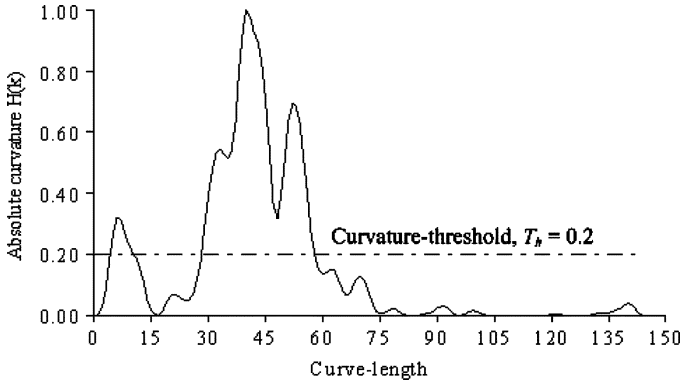

Fig. 8. Absolute curvature function of "curve 8" of "Lena" image [see Fig. 5(b)], where there is no prominent corners. However, all four curvature maxima points above the curvature-threshold are in the candidate corner set after the first refinement step in Section III-D1. We want to remove them using the second refinement step in Section III-D2.

Their localization is very good and curvature values are usually very high. In contrast, the weak corners are flat in nature and visually less significant features on curves. Their localization is very poor since it may be very difficult to point out their exact locations. Furthermore, their curvature values are usually smaller than the strong corners. The false corners are due to noise and the local variation on the curve. As we use small σ and the normalized discrete curvature values using (9) may be higher than the thresholds, some false corners may be detected. They are usually detected on the straight-line like curves (see curves 8 and 11 on "Lena" image in Fig. 5(b)) where there is no visually prominent corners. However, the curvature function of curve 8 of "Lena" image shown in Fig. 8 depicts that four false corners (four maxima points) are detected. The reason is that on a straight-line like curve the denominator of (9) will be relatively very small leading to high normalized curvature on some points.

We follow a two-step refinement process to obtain the final corner set by filtering out the weak and false corners. The first refinement step uses a curvature-threshold mainly to remove the weak corners and the second refinement step uses an angle-threshold mainly to remove the false corners.

1) Using Curvature-Threshold: The absolute curvature of a strong corner should be greater than that of a weak corner. The existing CSS-based detectors either used one [8], [10] or three [3], [9] predefined thresholds or adaptive (curve dependant) thresholds [11] to remove weak corners.

In our case, the absolute curvature on each point becomes within the range [0,1] due to normalization. Consequently, a single curvature-threshold T_h is sufficient to remove the weak corners. By experiments, we found that $T_h = 0.2$ works fine. If a local maximum is less than T_h, this maximum point is declared as a weak corner and removed from the candidate corner set.

2) Using Angle-Threshold: Fig. 8 shows that the false corners detected on a straight-line like curve may not be removed using the first refinement step discussed above, because their curvature values are higher than the threshold T_h. In the second refinement step, we estimate the angle at each candidate corner point using two curve-segments between this candidate and two of its nearest neighbor candidates on its both sides. If the estimated angle is larger than the angle-threshold, the candidate

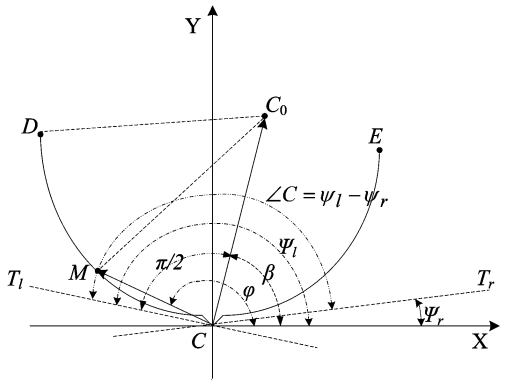

Fig. 9. Angle detection on the candidate corner C.

corner is removed from the candidate corner set. Note that this corner detection is different from the original definition of the curvature (see (1) in Section II-A), which estimates the rate of local angle change at each point with respect to the arc-length. Since we consider the curve-segments for each candidate corner point while estimating the angle, this makes a global sense of the estimated angle on the curve.

We know that a well defined corner should have a relatively sharp angle [11]. If we know the angle of each corner on a curve, it would be easy to differentiate between strong and false corners [16]. We follow the angle calculation technique similar to a technique in [11] as shown in Fig. 9.

The angle $\angle C, 0 \leq \angle C \leq \pi$, at a candidate corner C is estimated using the two tangents T_l and T_r on both sides of C. The *region-of-support* (RoS) for $\angle C$ is defined using two nearest candidate corners D and E on both sides of C. $D$and E could be the end points of Γ_s if no neighbor candidate corners are found on the same curve. In order to determine the left tangent angle $\psi_l, 0 \leq \psi_l \leq 2\pi$, a simple three-point method is employed to fit a circle approximately on the left "arm" CD of the RoS. The two points on the left arm CD are two end points C and D of the arm and the third point is their midpoint M. If these three points are collinear then the tangent direction on the left side of C is from C to D. Otherwise, the center C_o of the circle, which goes through points C, M, and D, is calculated. Let β, $0 \leq \beta \leq 2\pi$, be the angle representing the direction of C to C_o and $\varphi, 0 \leq \varphi \leq 2\pi$, be the angle representing the angle from C to M. Then we have the left tangent angle ψ_l on the left arm of C as

$$\psi_l = \beta + \text{sign}(\sin(\varphi - \beta))\frac{\pi}{2}. \tag{11}$$

where the function $\text{sign}(\cdot)$ returns the sign of its argument. Similarly, the right tangent angle $\psi_r, 0 \leq \psi_r \leq 2\pi$, on the right arm CE is calculated. Therefore, the angle $\angle C$ can be obtained as

$$\angle C = \begin{cases} |\psi_l - \psi_r|, & \text{if } |\psi_l - \psi_r| < \pi, \\ 2\pi - |\psi_l - \psi_r|, & \text{otherwise.} \end{cases} \tag{12}$$

Finally, C is removed from the candidate corner set if $\angle C$ satisfies the following condition

$$\angle C > \delta, \tag{13}$$

where δ is the maximum angle below which the estimated corner angles are considered as sharp angles. In our experiments the angle-threshold δ was set 157°.

E. Final Corner Set

The candidate corner set after removing weak and false corners is obtained as the final corner set. There may be a corner between the ends of a loop curve and such corner may be missed in spite of the curve extension. To detect such corner, we estimate the angle at any end point of the smoothed loop curve (Γ_s) using the technique shown in Fig. 9, where D and E are two already detected corners nearest to both ends. We add such end point of a loop curve to the final corner set if the estimated angle does not satisfy (13) and the end is far away from the detected corners. At last, a T-corner is added to the final corner set if it is far away from the already detected corners. In both the above cases, we consider a 5×5 neighborhood [8]. Fig. 12(d) shows the detected corners by the proposed CPDA detector in original "Lena" image.

IV. Performance Study

In this section, we present the outcome of two main performance studies. First, we summarize the experimental results of the parameter setting for the proposed CPDA detector (Section IV-C1). Second, we compare the performance of the proposed CPDA corner detector (Section IV-C2) with four existing most promising CSS corner detectors namely i) CSS detector [8], ii) ARCSS detector [3], iii) MSCP detector [10], and iv) He and Yung detector [11]. We also compare the proposed detector with the well known k-cosine technique [16] (RJ73). The performance of the other existing detectors (e.g., ECSS detector [9]) has already been proven to be inferior to some of the above detectors (e.g., CSS [8] and ARCSS [3] detectors) in [3]. In the comparative study, all the detectors were set with their default parameter values.

In our experiments, all detectors were executed using the same edge extraction and selection step, discussed in Section III-A, for fair comparisons. Moreover, for the RJ73 detector [16] the edges were smoothed using small Gaussian smoothing scales before corner detection (like we do for the proposed CPDA detector). In fact, the RJ73 did not discuss about smoothing to eliminate noise and we observed that the pre-smoothing increased the performance of this detector. We detected corners both in the original images and their attacked (geometric transformed and signal processed) images. Then we obtained the number of repeated corners between original and attacked (test) images. In the case of geometric transformations, we transformed the original corners using the transformation parameters prior to finding their repetitions in the test (transformed) corner set. This technique of performance study is automated and was proposed in [3]. Thus we avoid the traditional evaluation technique based on the *human judgement* which is hard to implement for a large database having thousands of test images [3]. Instead, we evaluated the robustness of the detected corners by using two metrics—*average repeatability* and *localization error* [3].

TABLE I
PARAMETER SETTING AT DIFFERENT STAGES BY THE PROPOSED CPDA CORNER DETECTOR

Stages	Parameters	Reasons	How decided
Edge extraction and selection	$low = 0.2$, $high = 0.7$	Control number of edges	Experimentation
"	$\alpha = 25$	Control length of edges	Source Code of [11]
"	$gap1 = 2$ pixel	Detect T-corners	[8]
"	$gap2 = 5$ pixel	Detect curve type	Source Code of [11]
Curve smoothing	$\sigma = 1, 2, 3$	Eliminate noise	Experimentation
Curvature estimation	$L = 10, 20, 30$	Capture different features	Experimentation
Corner set refinement	$T_h = 0.2$	Remove weak corners	Experimentation
"	$\delta = 157^\circ$	Remove false corners	Experimentation
Final corner set	5×5 neighborhood	Consider T-corner	[8]

A. The Database

We had total 23 different original 512×512 gray-scale images including some artificial images like "Block" and real world images like "Lena," 'Leaf," "House," and "Lab." Many of the above original images were collected from standard databases [17], [18] and can be found with the detected corners in [19]. We had their total 8694 transformed images as test images, which were obtained by applying the following seven different types of attacks on each original image:

- *Rotation*: 18 different angles θ in $[-90^\circ, +90^\circ]$ at 10° apart, excluding 0°.
- *Uniform* (U) *scale*: scale factors $s_x = s_y$ in [0.5, 2.0] at 0.1 apart, excluding 1.0.
- *Non-uniform* (NU) *scale*: scale factors s_x in [0.7, 1.5] and s_y in [0.5, 1.8] at 0.1 apart, excluding the cases when $s_x = s_y$.
- *Combined transformations* (*rot.-scale*): θ in $[-30^\circ, +30^\circ]$ at 10° apart, excluding 0°, followed by uniform or non-uniform scale factors s_x and s_y in [0.8,1.2] at 0.1 apart.
- *Lossy JPEG compression*: compression at 20 quality factors in [5, 100] at 5 apart.
- *Gaussian* (G) *noise*: zero mean white Gaussian (G) noise at 10 variances in [0.005,0.05] at 0.005 apart.
- *Shearing*: shear factors sh_x and sh_x in [0,0.012] at 0.002 apart, excluding $sh_x = sh_y = 0$.

Therefore, we had total 414 rotated, 345 uniform scaled, 2691 non-uniform scaled, 3450 rotated and scaled transformed images. We also had 460 JPEG compressed, 230 Gaussian noised, and 1104 sheared images. Note that transformations comprising rotations were also followed by cropping such that the outer black parts were disappeared. Consequently, many detected corners in the original images were cropped off in the test images for the transformations involving rotations.

B. Evaluation Metrics

We measure the performance of a corner detector in terms of the robustness of the detected corners. We can easily find the repeated corners between the original and signal processed (JPEG compressed and Gaussian noised) images. However, for geometric transformed images, we apply the original transformation to the original corner set prior to finding their repetitions in the test corner set.

The traditional technique of robustness evaluation involves *human visual inspection* procedure which is hard to implement for proper robustness tests due to following reasons [3]. First, it is very hard to point out all corner locations in natural images by human intuition. Second, human eyes are unable to measure the corner strength. Third, the volume of works with a large image database for ground truth collection prohibits its adoption. Finally, there is no standard procedure to collect ground truth, e.g., how many people should be involved and how to assess their decisions etc.

The automatic robustness evaluation previously proposed in [3] involves no human involvement and can be used with any size of database. This evaluation technique uses two metrics *average repeatability* and *localization error* together to measure the robustness of the detected corners. The average repeatability R_{avg} measures the average number of repeated corners between original and test images. It is defined as

$$R_{\text{avg}} = \frac{N_r}{2}\left(\frac{1}{N_o} + \frac{1}{N_t}\right) \tag{14}$$

where N_o and N_t are numbers of corners detected in the original and test images respectively and N_r is the number of repeated corners between them.

The localization error L_e is defined as the amount of pixel deviation of a repeated corner. It is measured as the *root-mean-square-error* (RMSE) of the repeated corner locations in the original and test images [3]:

$$L_e = \sqrt{\frac{1}{N_r}\sum_{i=1}^{N_r}(x_{oi} - x_{ti})^2 + (y_{oi} - y_{ti})^2} \tag{15}$$

where (x_{oi}, y_{oi}) and (x_{ti}, y_{ti}) are the positions of i-th repeated corner in the original and test images respectively. An RMSE value of maximum 3 pixels was allowed to find a repetition.

C. Experimental Results

In this section, we present two main parts of the experimental results. We first summarize the experimental results of the parameter setting for the proposed CPDA detector. We then present the comparative results between the proposed and existing detectors.

1) Summary of CPDA Parameter Setting: Table I shows the parameters which the CPDA detector uses at its different stages. Many of the parameter values were decided by experimentation. Below we summarize the experimental results and discuss how they were decided.

Fig. 10 shows the effect of Canny edge detection thresholds changes on the CPDA corner detector. When both the *low* and *high* thresholds were set small, the average number of detected corners by the CPDA detector was quite high, but its robustness

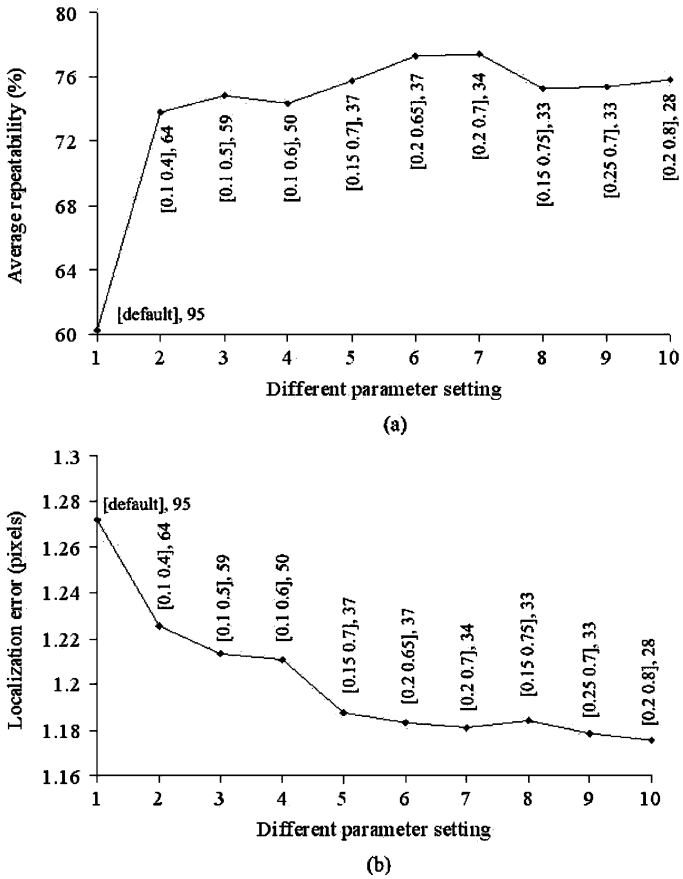

Fig. 10. Effect of Canny edge detection thresholds changes on the CPDA corner detector: (a) average repeatability and (b) localization error. At each data label the values within the square brackets are *low* and *high* thresholds of Canny edge detector and the value outside the bracket is the average number of detected corners by the CPDA detector.

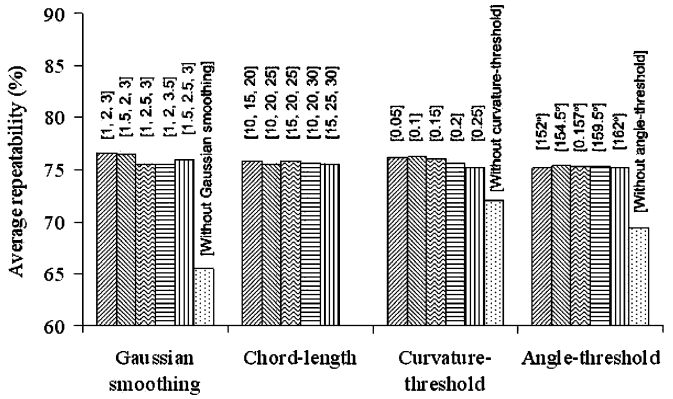

Fig. 11. Effect of different parameter changes (Gaussian smoothing scale, chord-length, curvature-threshold, and angle-threshold) on the CPDA corner detector. For each parameter, we had 5 different sets of values and we showed them in the square brackets above the corresponding bar.

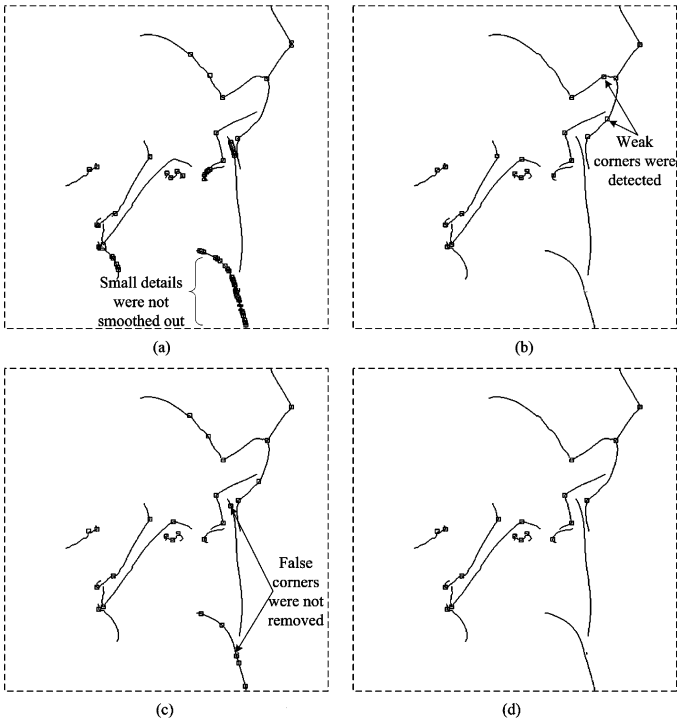

Fig. 12. Corner detection examples on "Lena" image by the proposed CPDA detector under different conditions: (a) without inital Gaussian smoothing, (b) without applying curvature-threshold, (c) without applying angle-threshold and (d) after all stages (final corner detection).

was low (low average repeatability and high localization error). The reason is, at lower thresholds the Canny edge detector obtains too many edges, many of which are weak and/or noisy. In contrast, when the thresholds were increased, only strong edges were detected; therefore, the robustness of the CPDA detector increased. However, when the thresholds were increased above $low = 0.2$ and $high = 0.7$, the robustness (average repeatability) of the CPDA detector decreased. Therefore, we have chosen the Canny edge detection thresholds at $low = 0.2$ and $high = 0.7$ as default for the CPDA detector when it offered the highest robustness (maximum repeatability and lower localization error).

Fig. 11 shows the effect of different parameter changes (Gaussian smoothing scale σ, chord-length L, curvature-threshold T_h, and angle-threshold δ in Table I) on the CPDA corner detector. We considered five different sets of values for each parameter. We observed that for a particular parameter, the average repeatability of the CPDA detector did not change much. Therefore, for each parameter we have selected the parameter value(s) producing the least localization error as default parameter value(s).

Fig. 11 also shows that the average repeatability was quite low without using different parameters at different stages of the proposed CPDA detector. Fig. 12(a) shows that without initial Gaussian smoothing the detector found too many false corners on some curves. Fig. 12(b) shows that some of the weak corners were detected without applying the curvature-threshold. Fig. 12(c) shows that a few false corners were detected without applying the angle-threshold. Finally, Fig. 12(d) shows the corner detection with all the parameter settings at their default values as given in Table I.

2) Comparative Results: Fig. 13 shows the average repeatability and localization error under different geometric transformations, JPEG lossy compression, and Gaussian noise. In general, all the detectors offered the highest (i.e., best) average repeatability in JPEG compression, but the highest localization

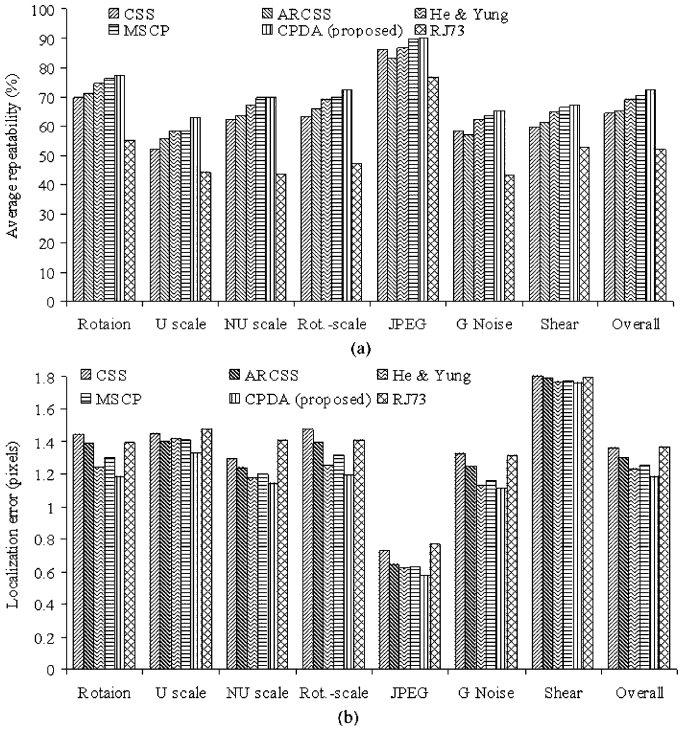

Fig. 13. Detector's robustness: (a) average repeatability and (b) localization error.

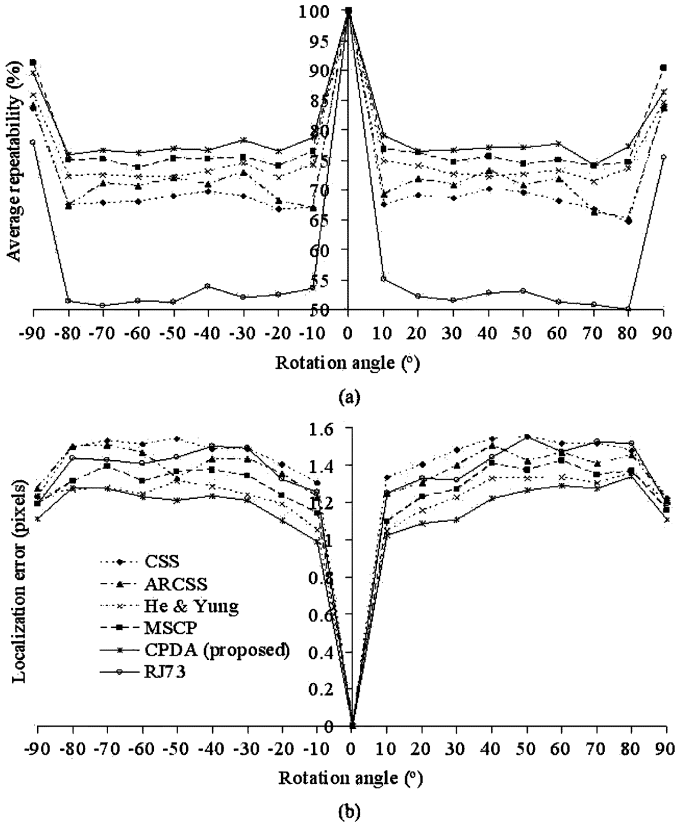

Fig. 14. (a) Average repeatability and (b) localization error under rotation.

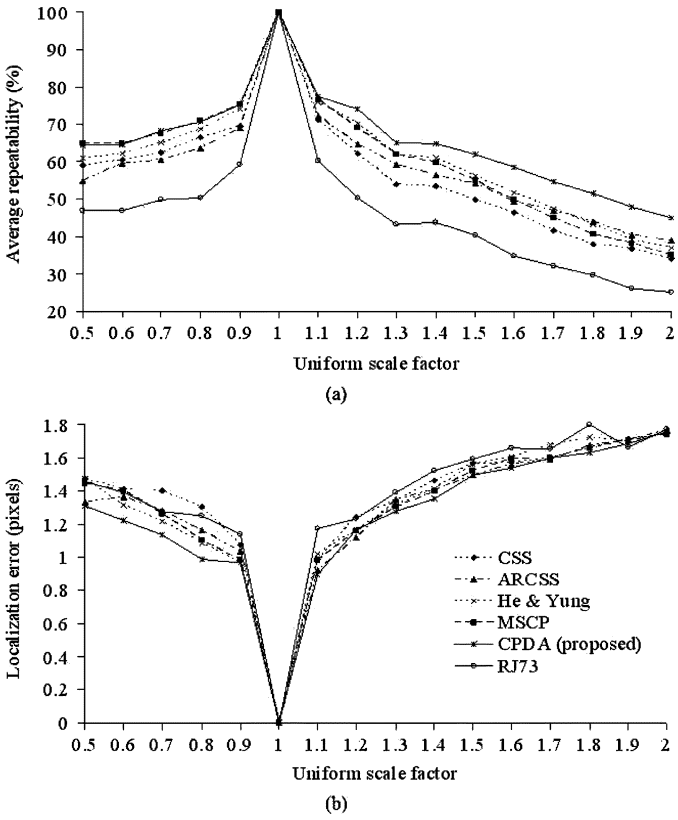

Fig. 15. (a) Average repeatability and (b) localization error under uniform scale change.

error (i.e., worst) in shearing. The proposed CPDA corner detector offered higher average repeatability and better localization (lower error) than all the existing detectors. Among the existing detectors, the MSCP detector offered the highest average repeatability. However, it suffered from localization error; since, unlike its parent the CSS detector, it did not execute any corner localization step. The ARCSS detector performed better than the CSS detector in geometric transformations as expected. However, in signal processing attacks (JPEG and noising), the ARCSS detector offered lower average repeatability but better localization than the CSS detector. The reason is the former (ARCSS) detected corners in lower scales than the latter (CSS).

The k-cosine based detector [16] showed the least average repeatability among all the detectors. The reason is, its smoothing scale depends on the curve-length ($n/10$, where $n =$ curve-length). Consequently, it always chooses a different scale under geometric transformations (scaling, shearing) when the curve-length is affected, even in other attacks (rotation, JPEG, Gaussian noising) where the Canny edge detector may extract the same edges as the original image, but with different lengths.

These results are shown elaborately in Figs. 14–18. Fig. 14 shows that the robustness of each detector remained almost the same in all rotations, except in $\pm 90^{\circ}$ when no corners were cropped off in the transformed images. Fig. 15 shows that the robustness of each detector decreased gradually with the increase of scaling factors above 1.0 and with the decrease of scaling factors below 1.0. Fig. 16 shows that the robustness became better when the JPEG quality factor increased. Fig. 17 shows that the robustness decreased considerably with the increase of the noise variance. Fig. 18 shows the same scenario when the robustness became worse at high shear factors in both x and y-directions. Moreover, localization error was worst in shearing for all the detectors. Here in Fig. 18 we present the detailed shear results

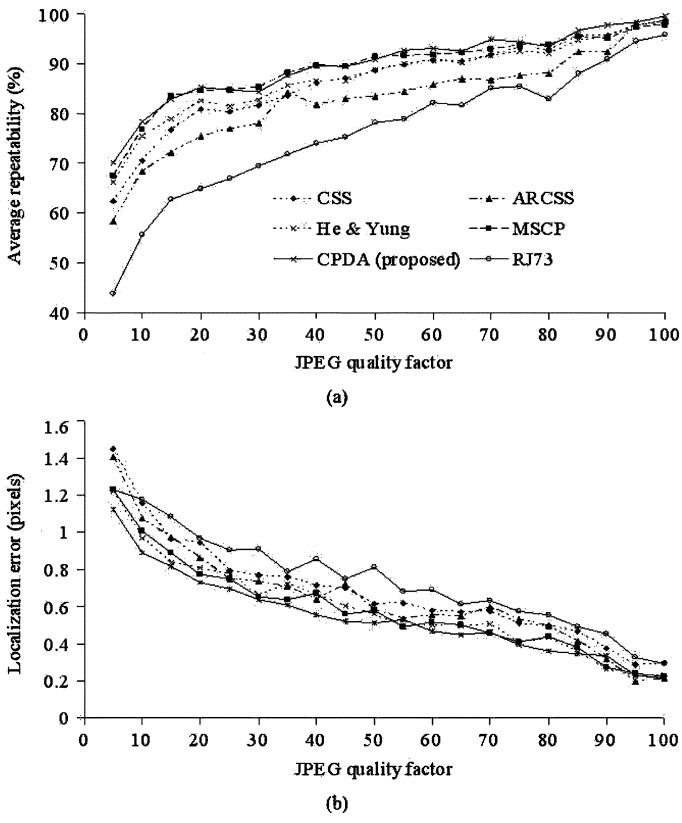

Fig. 16. (a) Average repeatability and (b) localization error under JPEG.

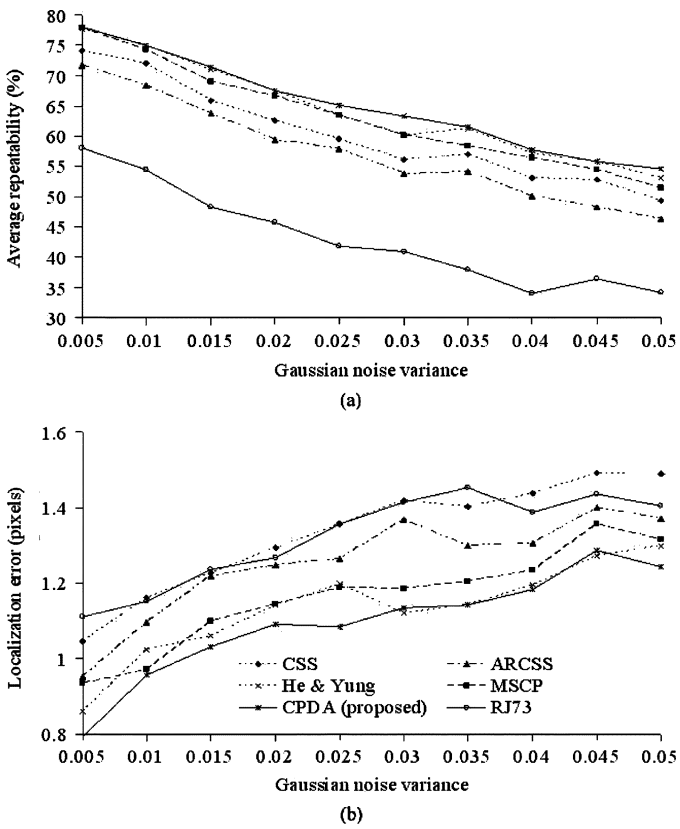

Fig. 17. (a) Average repeatability and (b) localization error under Gaussian noise.

for the CPDA detector only, which outperformed the existing CSS-based detectors in all shear factors considered in our experiments. The detail for other detectors can be found in [19].

D. Discussions

In this section, we first discuss how the proposed CPDA detector overcomes the key problems with the existing CSS-based detectors and then we will analyze the improved performance by the CPDA detector with the help of an example.

1) CPDA Overcomes CSS Key Problems: We have discussed two key problems in Section II-B associated with the existing CSS corner detectors. First, the CSS technique directly implements the "curvature definition" whose digital version involves higher order derivatives of curve-point locations up to second order. Second, the difficulty with appropriate Gaussian smoothing-scale selection leads to the inappropriate smoothing-scale selection.

The proposed CPDA detector overcomes the two aforementioned problems greatly. First, the CPDA discrete curvature estimation does not directly implement the "curvature definition" and, therefore, it does not use any derivative of curve-point locations at all. This difference in curvature estimation makes the CPDA detector to be less sensitive to the local variation as well as to the noise on the curve than the existing CSS-based detectors. Fig. 19 shows that the CSS technique considers a very small neighborhood (see Section II-A) to calculate the derivative-based curvature which may be too high in a region with high local variation. In contrast, the CPDA technique considers a larger neighborhood based on the involved chord-length. In Fig. 19, "chord 1 (short)" accumulates (sums) both positive and negative distances and "chord 2 (long)" accumulates distances which are more or less almost the same for all points inside and outside this region. So the curvature product from the accumulated distances using three chords of different lengths will be almost the same for all the points in and out of the local variation region. As a result, the proposed corner detector detects corners with higher average repeatability.

Second, the CPDA discrete curvature estimation does not require the appropriate Gaussian smoothing-scale (σ) selection. Therefore, the CPDA detector smoothes curves with one of three low Gaussian smoothing-scales which offer better localization of the detected corners. In contrast, the Gaussian smoothing is an integrated part of the CSS curvature estimation as discussed in Section II-A. Consequently, the existing CSS-based detectors use comparatively higher σ values than the proposed CPDA detector. However, smoothing with high σ shrinks the curve resulting poor localization of the detected corners and may remove some prominent corners (see Fig. 3 in Section II-B).

However, the CPDA detector uses a second smoothing-scale which is the chord-length L. In Section IV-C1, we discussed that moderate changes to three medium L values do not affect the detector's performance much. The use of curvature-product makes strong corners more distinguishable than the false and weak corners. In contrast to the Gaussian smoothing, the chord smoothing does not change the curve-point locations. Our comprehensive experiments showed that the proposed CPDA detector performed better than the existing CSS-based detectors.

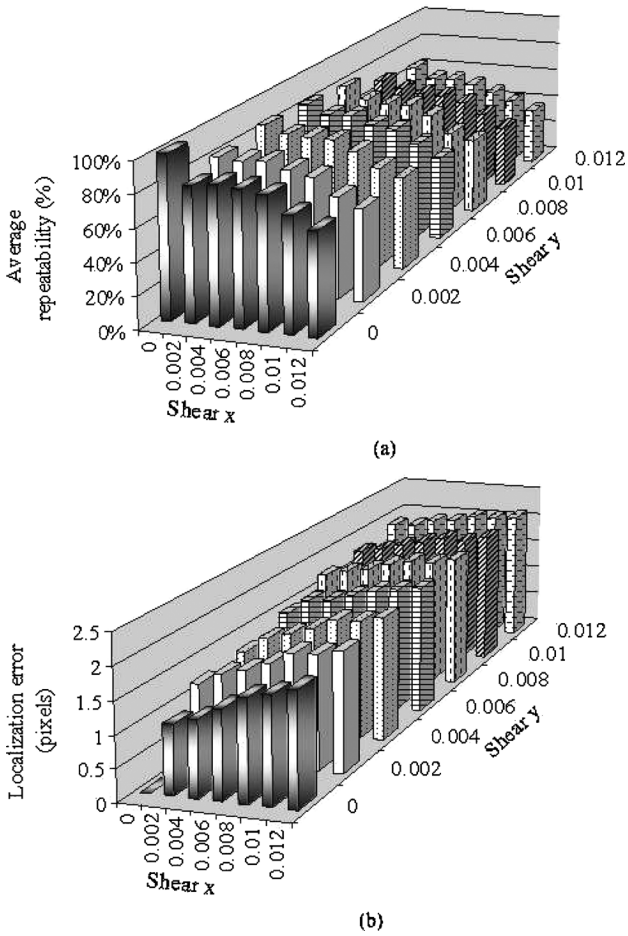

Fig. 18. Effect of Shearing to the proposed CPDA corner detector.

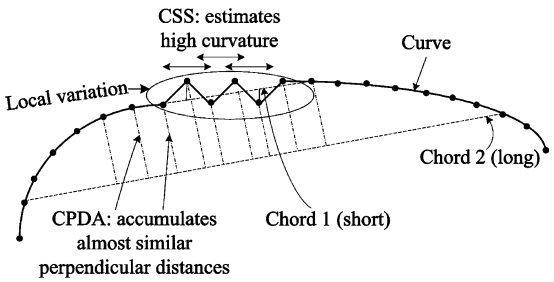

Fig. 19. Sensitivity of curvature estimation techniques to the high local variations or noise on the curve.

2) Analyzing the Improved Performance: The average repeatability and localization error reported in Section IV-C2 were averaged over many thousand corners. Many of these corners are quite stable, i.e., they can be detected by many existing detectors. However, there are some difficult cases where some corners may be missed and/or some weak/false corners may be detected. The difference between ours and other existing detectors is, our proposed CPDA detector performs better than others in such difficult situations. The improved averaged performance of the proposed corner detector is partly due to this difference.

Fig. 20 shows corner detection examples by different detectors in such a difficult situation. For each detector, we detect corners on an original long curve (inside large dotted rectangle) and on its 60% down-scaled version (inside small dotted rectangle).

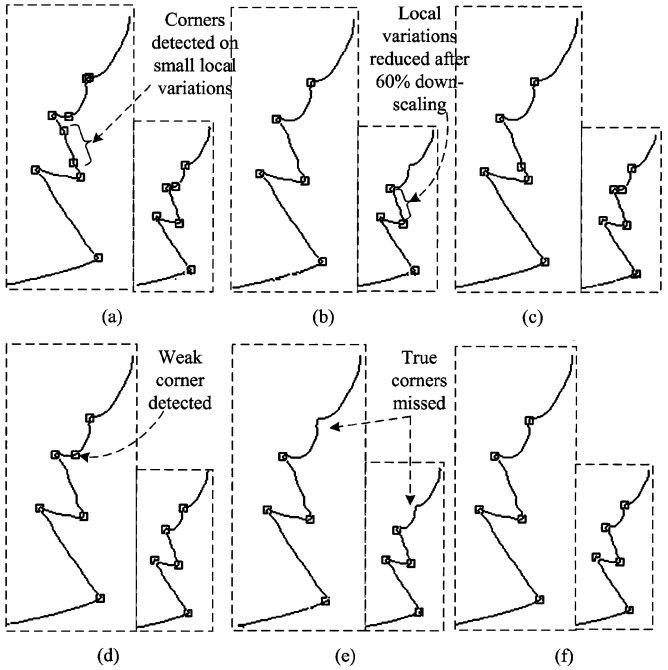

Fig. 20. Corner detection examples by different detectors. For each detector, we detect corners on an original long curve (inside large dotted rectangle) and on its 60% down-scaled version (inside small dotted rectangle). The curve in this example is from the "Leaf" image in [19].

On the original curve there were some local variations where the CSS and He & Yung detectors detected some false corners. On the down-scaled version when the local variations were reduced, these detectors did not detect them anymore. Some of the existing detectors also detected a weak corner either on the original curve (CSS and MSCP detectors) or on the down-scaled version (He and Yung). The ARCSS, the RJ73, and the proposed CPDA detectors detected neither the false nor the weak corners. However, the ARCSS detector missed a strong corner on the down-scaled version due to affine-length parameterization of the curve. The RJ73 also missed this strong corner on both the curves due to its large smoothing-scale (10% of the curve-length). However, in other cases we observed that the RJ73 detected many false and weak corners on small curves when the smoothing-scale was small. The proposed CPDA detector detected all the strong corners and did not detect any false or weak corners in this example.

V. Conclusion and Future Work

We have proposed a new corner detector based on the CPDA discrete curvature estimation technique [12]. It first smooths the extracted edges using the small scale Gaussian function to reduce the effect of noise. Then it detects corners on the smoothed curves in the chord scale-space using three different sizes of chords. The three normalized curvature values estimated on each point are multiplied which makes the strong corners more distinguishable from the weak corners. The maxima of the absolute curvature products are obtained as the candidate corner set from where weak and false corners are filtered out using the curvature-threshold and the angle-threshold respectively.

The proposed CPDA corner detector performs better than the existing CSS-based corner detectors in term of robustness. The

direct application of such a robust and stable corner detector is in the areas of the computer vision where a successful and effective image matching is a prerequisite. Other applications include geometric distortion correction in images [20], where precise localization of the repeated corners is vital for the accurate estimation of the geometric transformations between the images originated from the same original image. Future works may include more robust corner detectors and apply them in many of the computer vision research, e.g., image matching [4], geometric distortion correction [20], and mobile robot vision [21].

REFERENCES

[1] M. Awrangjeb and M. Murshed, "Robust signature-based geometric invariant copyright protection," in *Proc. IEEE Int. Conf. Image Processing*, Atlanta, GA, Oct. 2006, pp. 1961–1964.

[2] J. S. Seo and C. D. Yoo, "Image watermarking based on invariant regions of scale-space representation," *IEEE Trans. Signal Process.*, vol. 54, no. 4, pp. 1537–1549, Apr. 2006.

[3] M. Awrangjeb, G. Lu, and M. Murshed, "An affine resilient curvature scale-space corner detector," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Honolulu, HI, Apr. 2007, vol. 1, pp. 1233–1236.

[4] M. Awrangjeb and G. Lu, "A robust corner matching technique," in *Proc. IEEE Int. Conf. Multimedia & Expo*, Beijing, China, Jul. 2007, pp. 1483–1486.

[5] A. Rattarangsi and R. T. Chin, "Scale-based detection of corners of planar curves," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 14, no. 4, pp. 430–449, Apr. 1992.

[6] B. Zhong and W. Liao, "Direct curvature scale space: Theory and corner detection," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 29, no. 3, pp. 508–512, Mar. 2007.

[7] B. K. Ray and R. Pandyan, "ACORD-an adaptive corner detector for planar curves," *Pattern Recognit.*, vol. 36, pp. 703–708, 2003.

[8] F. Mokhtarian and R. Suomela, "Robust image corner detection through curvature scale space," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 20, no. 12, pp. 1376–1381, Dec. 1998.

[9] F. Mokhtarian and F. Mohanna, "Enhancing the curvature scale space corner detector," in *Proc. Scand. Conf. Image Analysis*, 2001, pp. 145–152.

[10] X. Zhang, M. Lei, D. Yang, Y. Wang, and L. Ma, "Multi-scale curvature product for robust image corner detection in curvature scale space," *Pattern Recognit. Lett.*, vol. 28, pp. 545–554, 2007.

[11] X. C. He and N. H. C. Yung, "Curvature scale space corner detector with adaptive threshold and dynamic region of support," in *Proc. Internatinal Conference on Pattern Recognition*, Cambridge, U.K., Aug. 2004, vol. 2, pp. 791–794.

[12] J. H. Han and T. T. Poston, "Chord-to-point distance accumulation and planar curvature: A new approach to discrete curvature," *Pattern Recognit. Lett.*, vol. 22, pp. 1133–1144, 2001.

[13] F. Mokhtarian and S. Abbasi, "Affine curvature scale space with affine length parametrisation," *Pattern Anal. Applicat.*, vol. 4, no. 1, pp. 1–8, 2001.

[14] T. Y. Phillips and A. Rosenfeld, "A method of curve partitioning using arc-chord distance," *Pattern Recognit. Lett.*, vol. 5, pp. 285–288, Apr. 1987.

[15] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 8, no. 6, pp. 679–698, 1986.

[16] A. Rosenfeld and E. Johnston, "Angle detection on digital planar curves," *IEEE Trans. Comput.*, vol. 22, pp. 875–878, 1973.

[17] F. A. P. Petitcolas, Photo Database 2007 [Online]. Available: http://www.petitcolas.net/fabien/watermarking/image_database/index.html, [Online]. Available

[18] The Usc-Sipi Image Database 2007 [Online]. Available: http://sipi.usc.edu/database/, Available: USC-SIPI. [Online]

[19] M. Awrangjeb, Image Database and Corner Detection 2007 [Online]. Available: http://personal.gscit.monash.edu.au/awran/images.html, [Online]. Available

[20] M. Awrangjeb, M. Murshed, and G. Lu, "Global geometric distortion correction in images," in *Proc. IEEE Int. Workshop on Multimedia Signal Processing*, Victoria, BC, Canada, Oct. 2006, pp. 435–440.

[21] G. Cao, J. Chen, and J. Jiang, "A novel local invariant descriptor adapted to mobile robot vision," in *Proc. American Control Conference*, Boston, MA, USA, June 2004, vol. 3, pp. 2196–2201.

Mohammad Awrangjeb (S'02–M'03–S'06–M'08) received the B.Sc. Engg. degree in computer science and engineering from Bangladesh University of Engineering and Technology (BUET) in 2002 and the M.Sc. degree in computer science from National University of Singapore (NUS) in 2004. He has submitted his Ph.D. thesis with Monash University, Australia, where he had been fully funded.

He joined NUS as a Research Scholar. He is currently a Research Fellow with the University of Melbourne, Australia. Before joining NUS, he had worked at University of Asia-Pacific, Dhaka as a Lecturer in Computer Science and Engineering Department and after finishing his M.Sc. He joined American International University-Bangladesh as a Lecturer in the Computer Science Department. His research interests includes feature detection and matching, digital watermarking, digital photogrammetry multimedia security, biometrics, and network security. He has published three journal papers and ten conference papers in these areas.

Guojun Lu (M'96–SM'05) received the B.Eng. degree in 1984 from Nanjing Institute of Technology (now South East University), China, in 1984 and the Ph.D. degree in 1990 from Loughborough University, U.K.

He is currently a Professor at Gippsland School of Information Technology, Monash University, Churchil, Victoria, Australia. He has held positions at Loughborough University, National University of Singapore, and Deakin University, after he obtained his Ph.D. in 1990 from Loughborough University and B.Eng. in 1984 from Nanjing Institute of Technology (now South East University). Guojun's main research interests are in multimedia communications and multimedia information indexing and retrieval. He has published over 120 refereed journal and conference papers in these areas and wrote two books Communication and Computing for Distributed Multimedia Systems (Artech House 1996), and Multimedia Database Management Systems (Artech House 1999).